SPECIFICATION AMENDMENTS

Please amend paragraph [0031] as follows, which does not include an embedded hyperlink. The amendment to paragraph [0031] removes the embedded hyperlink, as Examiner requested in the Final Office Action dated November 28, 2007.

[0031] A detailed description of the JMX specification can be found on the Sun Website which is, as of the time of this writing, **[[http://java.sun.com/products/ JavaManagement/]]** http://java.sun.com/products/ JavaManagement/ (currently version 1.2.1 Reference Implementation).

Applicants observed an inadvertent error in the Specification. Namely, the Figures illustrate a "central database 330," which is referred to throughout the specification as "central database 230." Applicants note that reference number 230 corresponds to "Web Browser 230" of Figure 2. Thus, Applicants respectfully request that the following amendments be entered into the Description. Specifically, amendments are made to paragraphs [0037], [0039], [0040], [0044], [0049], and [0055]-[0057] to change "database 230" to read --database 330--.

[0037] In one embodiment, the locking service 302 disables access to (i.e., locks) certain specified portions of configuration data and/or program code stored within a central database ~~230~~ **330**. A locking manager 340, 350 employed within the server nodes locks data on behalf of various system components which need to synchronize access to specific types of data and program code (e.g., such as the configuration managers 344, 354 illustrated in Figure 3). As described in detail below, in one embodiment, the locking service 302 enables a distributed caching architecture for caching copies of server/dispatcher configuration data.

[0039] As illustrated in Figure 3, each application server (e.g., 318, 328) includes a lock manager 340, 350 for communicating with the locking service 302; a cluster manager 342, 352 for communicating with the messaging service 204; and a configuration manager 344, 354 for communicating with a central database ~~230~~ **330** (e.g., to store/retrieve configuration data).

Although the lock manager 340, 350, cluster manager 342, 352 and configuration manager 344, 354 are illustrated with respect to particular server nodes, 318 and 328, in Figure 3, each of the server nodes 314, 316, 324 and 326 and/or on the dispatchers 312, 322 may be equipped with equivalent lock managers, cluster managers and configuration managers.

[0040] Referring now to Figure 4, in one embodiment, configuration data 420 defining the configuration of the central services instance 300 and/or the server nodes and dispatchers within instances 310 and 320, is stored within the central database ~~**230**~~ **330**. By way of example, the configuration data may include an indication of the kernel, applications and libraries required by each dispatcher and server; network information related to each dispatcher and server (e.g., address/port number); an indication of the binaries required during the boot process for each dispatcher and server, parameters defining the software and/or hardware configuration of each dispatcher and server (e.g., defining cache size, memory allocation, . . . etc); information related to the network management configuration for each server/dispatcher (e.g., as described below); and various other types of information related to the cluster.

[0044] At 502, the configuration data within the configuration cache 400 of application server 328 is modified. At 504, the cluster manager 352 broadcasts an indication of the modified data to the cluster manager 342 on server node 318 and the cluster manager of other server nodes (i.e., via the messaging service 304). At 506, the modifications to the configuration data are committed to the central database ~~**230**~~ **330**. At 508, the cluster manager 352 notifies the cluster manager 342 on server node 318 and the cluster managers of other server nodes of the central database update. In response, the configuration manager 344 invalidates the modified configuration data from its cache 401 and, at 512, loads the new configuration data from the central database ~~**230**~~ **330**. In one embodiment, the configuration manager 344 only downloads the portion of the configuration data which has been modified (i.e., rather than the entire set of configuration data). To determine whether it needs to update its configuration data, the configuration manager 344 compares the version of its configuration data with the version of the configuration data stored the central database.

[0049] In one embodiment, a central monitoring service 600 employed within the distributed configuration hides the clusterization of the various MBean servers and provides a unified view

of managed resources at the manager level 201. Specifically, monitor integration logic 601 associated with the central monitoring service 600 combines the monitoring data collected from each of the individual MBean servers 603, 611, 621 and generates an comprehensive, logical view of the monitored resources. The monitoring data may then be displayed on a visual administrator 630 and/or any other type of graphical user interface 631 (e.g., such as a standard Web browser). In one embodiment, the integration logic 601 combines the monitoring data based on monitor configuration information 640 (e.g., node layout, monitoring parameters, . . . etc) stored within the central database ~~230~~ **330**. As described below with respect to Figure 8a-b, in one embodiment, the monitor integration logic 601 includes monitor MBeans arranged in a logical monitor tree hierarchy.

[0055] One particular embodiment, illustrated in Figure 8a, employs two different types of MBeans to perform passive and/or active instrumentation functions: resource MBeans 802 (also referred to herein as "runtime" MBeans) and monitor MBeans 801. Resource MBeans, also referred to herein as "runtime" MBeans, are associated with the underlying system resources such as kernel resources, components, libraries, . . . etc. Monitor MBeans are generated by the central monitor service 600 and are mapped to the resource MBeans according to the monitoring configuration data 640 stored within the central database ~~230~~ **330**.

[0056] Figure 8b illustrates a monitor initialization process utilized within the architecture of Figure 8a. At 850, the J2EE components required to run the monitor architecture are started (e.g., the management interfaces). At 852, the components then install/initialize the administration service (or application) 805. The administration service is also illustrated and described below with respect to Figure 13. At 854, the administration service 805 installs the resource MBeans used to monitor specified resources within each node in the cluster. In one embodiment, the administration service 805 uses the monitor configuration data 640 within the central database ~~230~~ **330** to identify the resource MBeans to install. The administration service 805 may communicate with the central monitor service 600 throughout the initialization process to coordinate the retrieval of MBean data form the central database ~~230~~ **330** and to coordinate the mapping of resource MBeans to monitor MBeans (as described below).

[0057] At 856, the central monitor service 600 installs the monitor MBeans 801 based on the monitor configuration data 640 stored within the central database ~~230~~ **330**. In one embodiment,

the central monitor service 600 arranges the Monitor MBeans 801 within a hierarchical monitor tree 800, representing the logical relationships between the resources in each of the nodes in the cluster. As mentioned above, monitor information from the monitor tree 800 (or subsections thereof) may be displayed within a graphical visual administrator 630 or other user interface.